**Project 2 Writeup: Supervised Learning**

### 1. Classification vs Regression

*Q: Your goal is to identify students who might need early intervention - which type of supervised machine learning problem is this, classification or regression? Why?*

A: We’re trying to predict a discrete outcome, whether a student will graduate or not, instead of a continuous outcome (like GPA or SAT scores) so it makes sense to use a **classification**-based supervised machine learning algorithm instead of a regression-based one.

### 2. Exploring the Data

*Q: Can you find out the following facts about the dataset?*

A:

Total number of students: 395  
Number of students who passed: 265  
Number of students who failed: 130  
Number of features: 30  
Graduation rate of the class: 67%

### 3. Preparing the Data

See iPython Notebook

### 4. Training and Evaluating Models

*Q: Choose 3 supervised learning models that are available in scikit-learn, and appropriate for this problem.*

A:

1. Decision Trees
2. Support Vector Machines
3. Gaussian Naïve Bayes

Q: What are the general applications of this model? What are its strengths and weaknesses?

A:

**Decision Trees**

* Time Complexity: O(f\*n²log(n)) where f is the # of features and n is the sample size.
* Space Complexity: O(f\*n\*log(n))
* General Application: Decision Trees make a prediction about a variable’s value given particular values for a set of features by following a series of “if-then-else” rules about the data.
* Strengths:
  + The series of rules used to classify the data can be visualized as a “tree” which makes it possible for a non-technical audience to follow the logic behind the model.
  + Can be validated using statistical tests.
  + Quick predictions that run in O(log(n)) time, where n points were used to train the tree.
* Weaknesses:
  + Unstable because variations in data can cause a completely different tree to be generated (which also leads to overfitting.)
  + Problem is NP complete if an optimal tree is desired.
  + Some problems like XOR are not easily expressed by a tree and may require exponential computing time to arrive at a reasonable answer.
* (Source: <http://scikit-learn.org/stable/modules/tree.html>)

**Support Vector Machines (SVMs)**

* Time Complexity: O(f\*n³), where f is the # of features and n is the sample size.
* Space Complexity: O(f\*n²)
* General Application: SVMs (when used as SVCs for classification) are trained by representing a set of points in m-dimensional space (where m is the # of features) and projecting those points into (m+1)-dimensional space to find hyperplanes that separate the data into different categories.
* Strengths:
  + Relatively effective when the # of samples is limited.
  + Can use custom kernel functions that are better tailored to the data to map those points into higher dimensional space.
* Weaknesses:
  + Computationally inefficient (cubic time)
  + Prone to overfitting
* (Source: <http://scikit-learn.org/stable/modules/svm.html>)

**Gaussian Naïve Bayes**

* Training complexity: O(f\*n), where f is the # of features and n is the sample size.
* Space complexity: O(f\*n)
* General Application: Naïve Bayes assumes all of the features are independent of each other and the likelihood of each feature is assumed to be Gaussian (other variants may assume the features follow a Multinomial or Bernoulli distribution.) For every outcome observed in the training set, it calculates a probability for each possible value for each feature while training. And it then uses this info to predict the most likely outcome for a given set of features and their values.
* Strengths:
  + Extremely fast
  + Don’t need a lot of data to train
* Weaknesses:
  + Assumes all features are independent; can be inaccurate when 2 or more features are highly dependent.
  + Doesn’t have very many parameters to tune which means you’d have to try another model if you’re not getting good accuracy.
* (Source: <http://scikit-learn.org/stable/modules/naive_bayes.html> )

**Benchmarks:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Decision Trees** | **Training Size 100** | **Training Size 200** | **Training Size 300** |
| **Training time (s)** |  |  | 0.003 |
| **Prediction time (s)** |  |  | 0.001 |
| **F1 Score for training** |  |  | 1.0 |
| **F1 Score for test set** |  |  | 0.809160305344 |

|  |  |  |  |
| --- | --- | --- | --- |
| **SVM (SVC)** | **Training Size 100** | **Training Size 200** | **Training Size 300** |
| **Training time (s)** |  |  | 0.014 |
| **Prediction time (s)** |  |  | 0.009 |
| **F1 Score for training** |  |  | 0.914414414414 |
| **F1 Score for test set** |  |  | 0.794701986755 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Gaussian NB** | **Training Size 100** | **Training Size 200** | **Training Size 300** |
| **Training time (s)** |  |  |  |
| **Prediction time (s)** |  |  |  |
| **F1 Score for training** |  |  |  |
| **F1 Score for test set** |  |  |  |